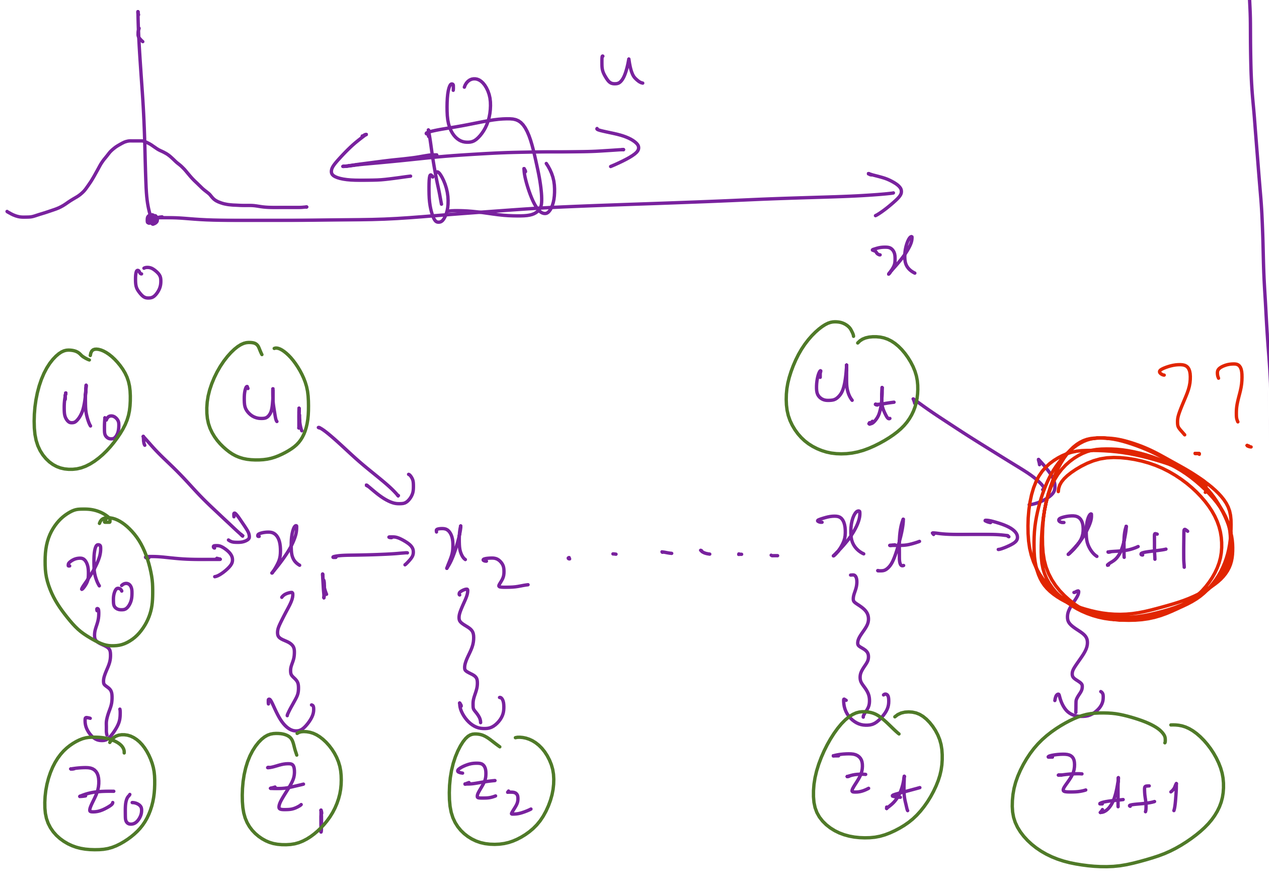
Introduction to Robot Intelligence
[Spring 2023]

Introduction to Planning + leftovers from filtering

April 18, 2023

Lerrel Pinto

Prior Model



$x_0 \sim N(0, \sigma_0)$ — prior model

$x_{t+1} \sim N(x_t + u_t, \sigma_R)$ - motion model

$z_{t+1} \sim N(x_{t+1}, \sigma_Q)$ — observation model

Question:

$$p(x_{t+1} \mid x_0, u_{0:t}, z_{0:t+1})$$

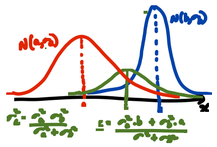
$$Bel(x_{t+1})$$

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

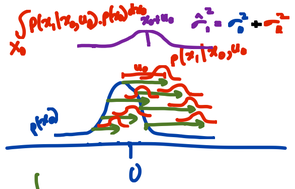
Inference

$$\underline{Bel}(x_{t+1}) = \eta \underbrace{p(z_{t+1} \mid x_{t+1})}_{\text{Obs. model}} \cdot \int_{x_t} \underbrace{p(x_{t+1} \mid x_t, u_t)}_{\text{Motion model}} \cdot \underbrace{p(x_t \mid x_0 \ldots u_{0:t+1} z_{0:t})}_{\text{Belief}(x_t)} dx_t$$

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf



Inference with 'Kalman' Gain



$$Bel(x_1) = \eta \; p(z_1|x_1) \cdot \int_{x_0} p(x_1|x_0,u_0) \cdot p(x_0)\, dx_0$$

$$p(z_1|x_1) \sim N(x_1, \sigma_R)$$

$$p(x_1|x_0,u_0) \sim N(x_0+u_0, \sigma_R)$$

$$p(x_0) \sim N(0, \sigma_0)$$

$$\hat{x}_1 = x_0 + u_0$$

$$\hat{\sigma}_1^2 = \sigma_0^2 + \sigma_R^2$$

Prediction step

$$N(\hat{x}_1, \hat{\sigma}_1)$$

$$N(\tilde{x}_1, \tilde{\sigma}_1)$$

$$\tilde{x}_1 = \frac{x_1 \cdot \hat{\sigma}_1^2 + \hat{x}_1 \cdot \sigma_R^2}{\hat{\sigma}_1^2 + \sigma_R^2}$$

$$\tilde{\sigma}_1^2 = \frac{\sigma_R^2 \hat{\sigma}_1^2}{\sigma_R^2 + \hat{\sigma}_1^2}$$

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

Inference with 'Kalman' Gain

$Bel(x_{t-1}) \rightarrow \mathcal{N}(\mu_{t-1}, \Sigma_{t-1})$ $\quad Bel(x_t)$ $\quad \mu_t, \Sigma_t$

$$\text{KalmanFilter}(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$$

prediction mean (motion, control, 'old' mean)

$$\bar{\mu}_t = A_t \mu_{t-1} + B u_t$$

prediction covariance ('old' covariance, Gaussian noise)

$$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^\top + R$$

Prediction ✓

$$K_t = \bar{\Sigma}_t C_t^\top (C_t \bar{\Sigma}_t C_t^\top + Q_t)^{-1}$$

Gain

update mean (observation model)

$$\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$$

update covariance

$$\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$$

Update

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

Inference with ‘Kalman’ Gain

Simple Multi-dimensional Implementation
(also 5 lines of code!)

```
[x P] = KF(x,P,z)

  x = A*x;
  P = A*P*A' + Q;

  K = P*C'/(C*P*C' + R);

  x   = x + K*(z - C*x);
  P   = (eye(size(K,1))-K*C)*P;
```

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

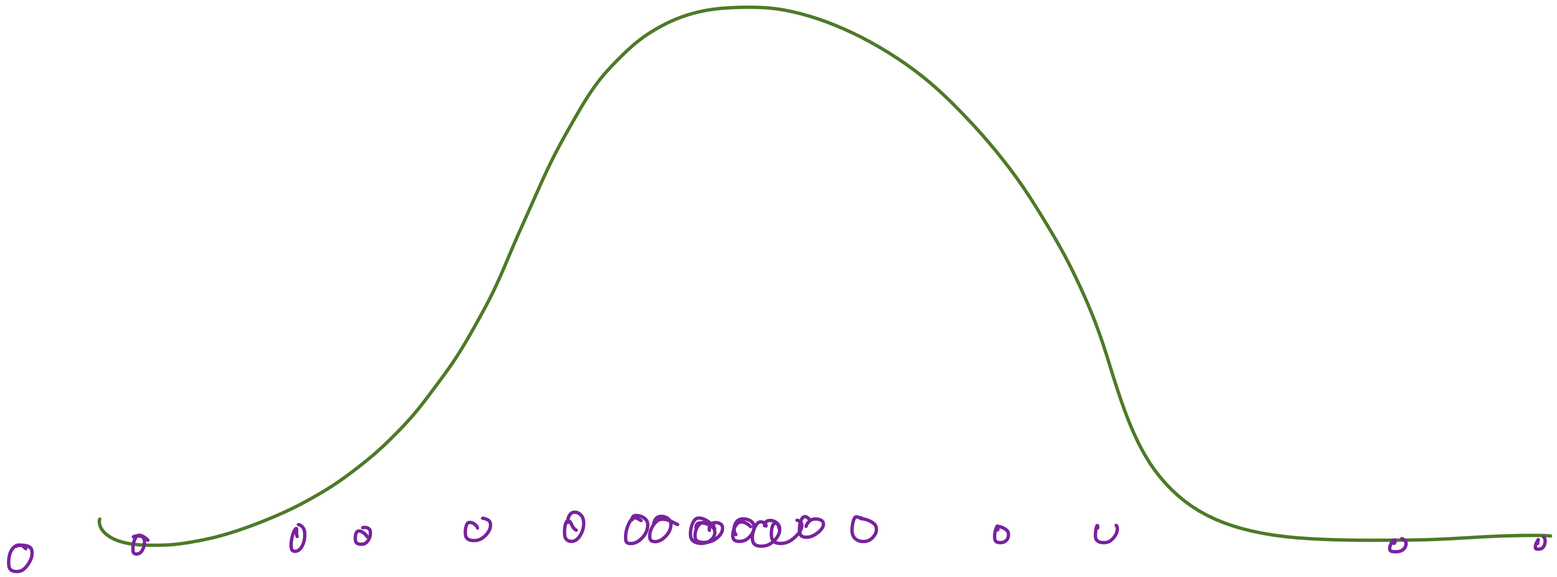
What do you do with non-linear dynamics?

EKF

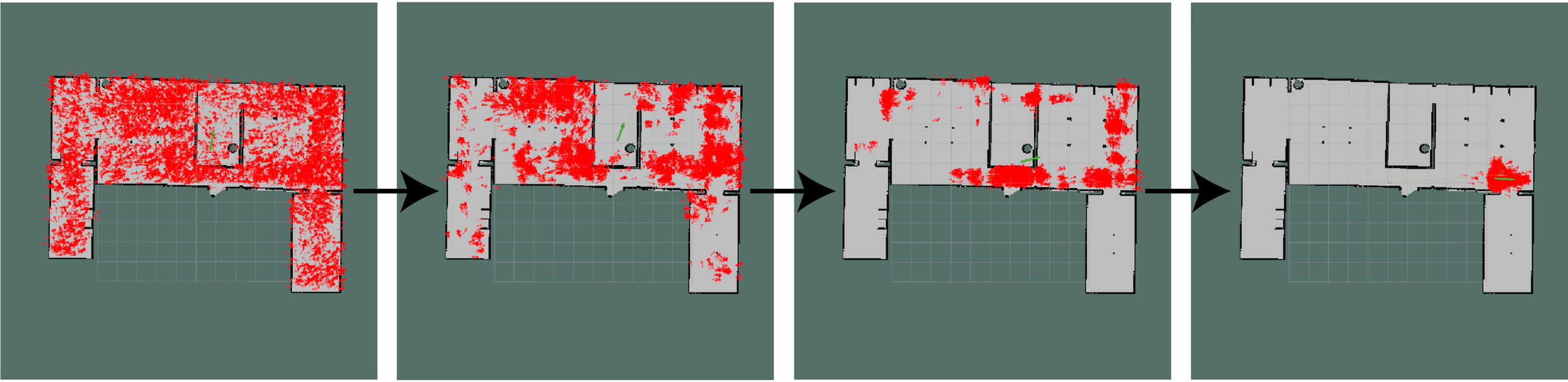
Extended Kalman Filter

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

More principled approach: Partile Filter



More principled approach: Partile Filter

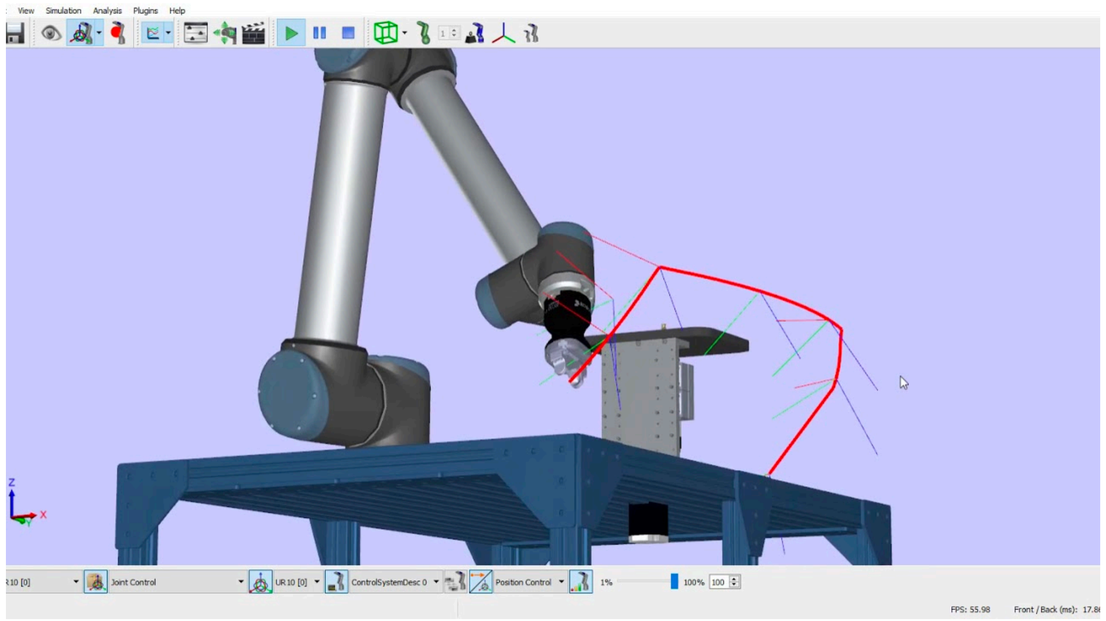


https://classes.cs.uchicago.edu/archive/2021/spring/20600-1/particle_filter_project.html

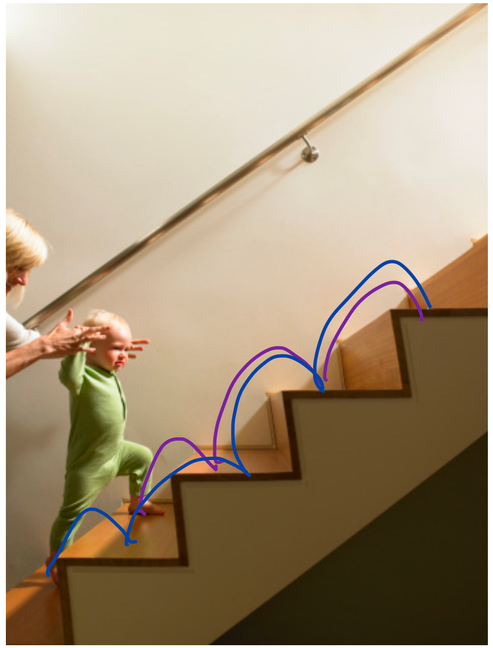
Robot Planning!!

2D Roomba example

Higher dimensional examples

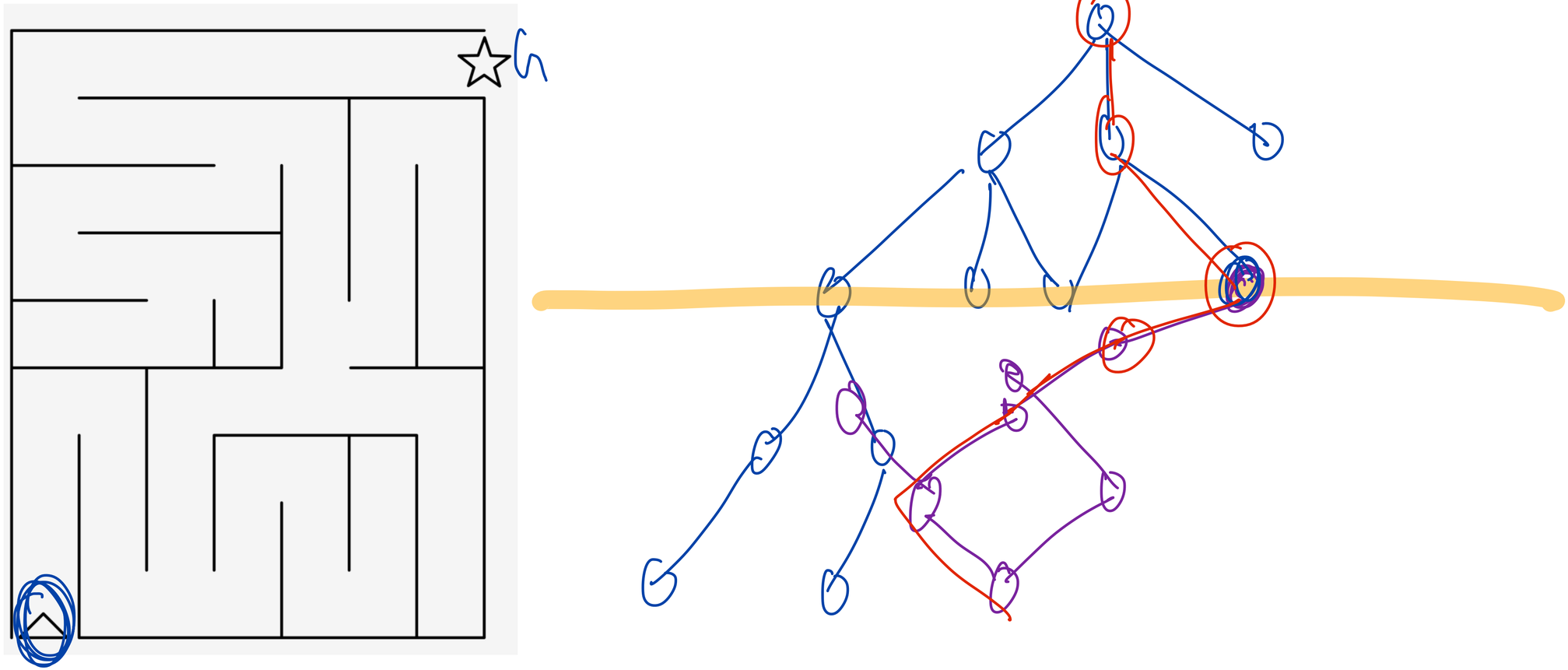


Higher dimensional examples

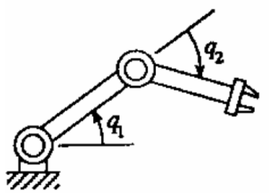


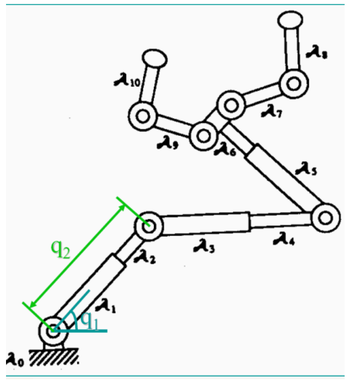
Even higher dimensional examples

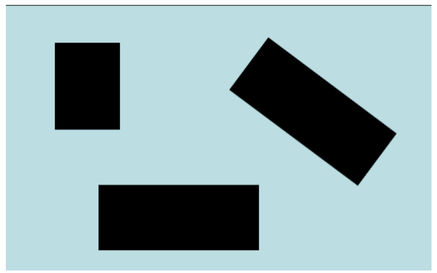
A 2D maze solving problem



Fundamental Concept: C-space



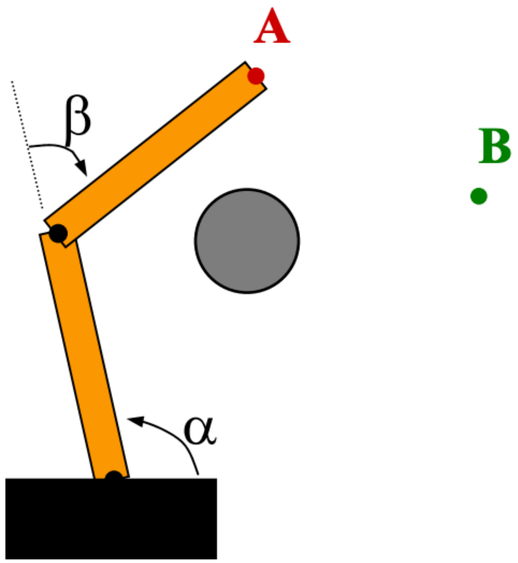




Configuration Space

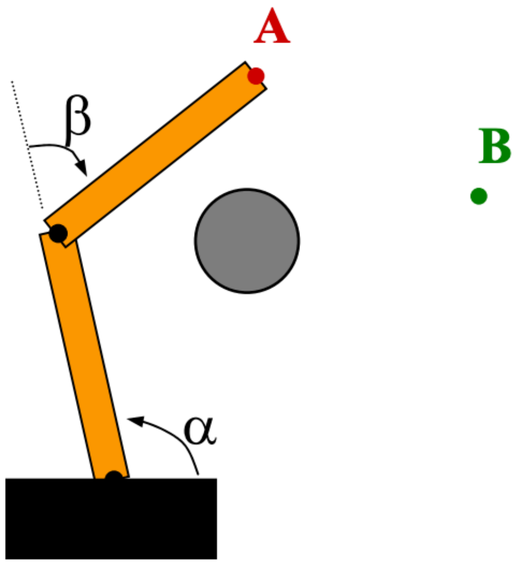
Planning Algorithms

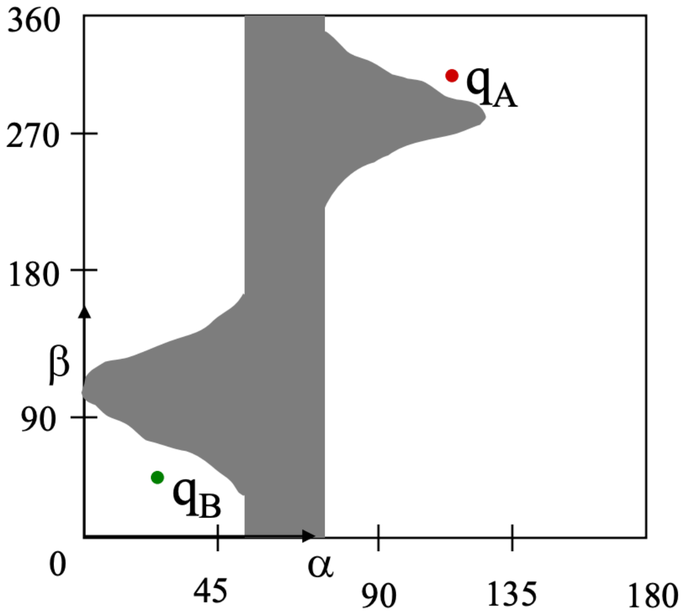
Fundamental Concept: C-space



Howie Choset with slides from G.D. Hager, Z. Dodds, and Dinesh Mocha

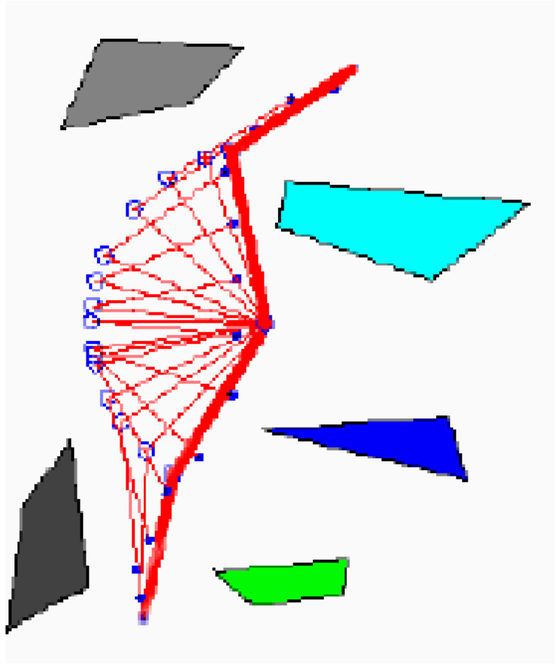
Fundamental Concept: C-space





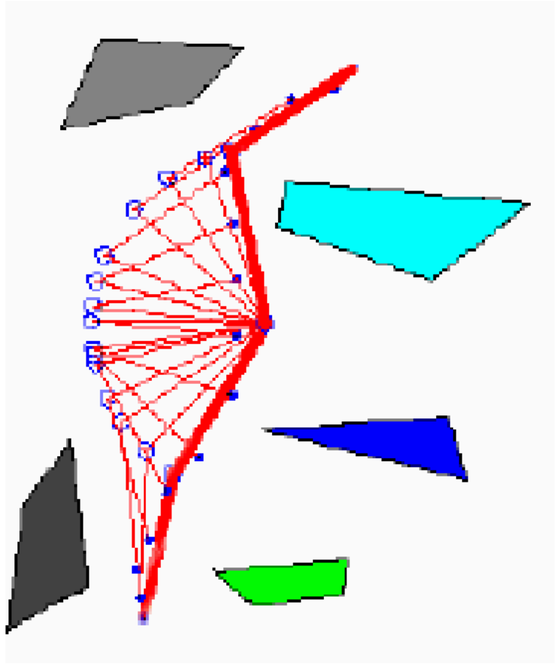
Howie Choset with slides from G.D. Hager, Z. Dodds, and Dinesh Mocha

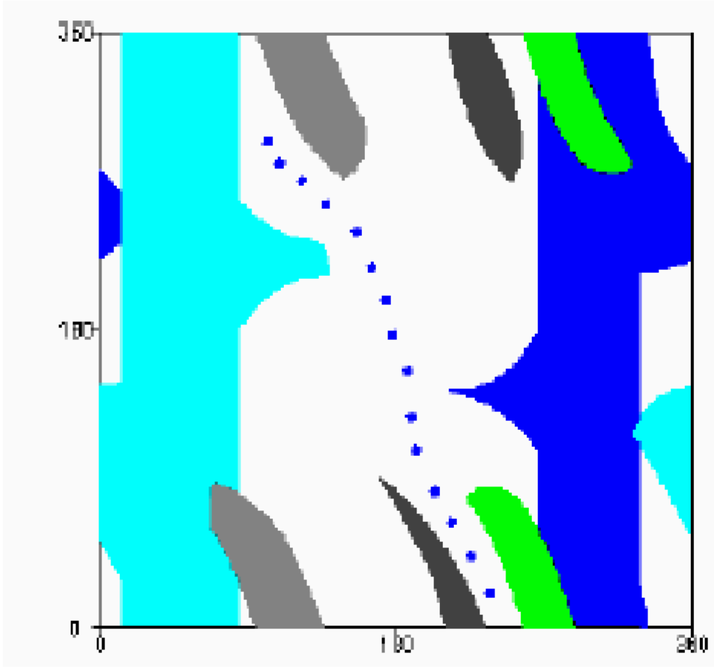
Fundamental Concept: C-space



Howie Choset with slides from G.D. Hager, Z. Dodds, and Dinesh Mocha

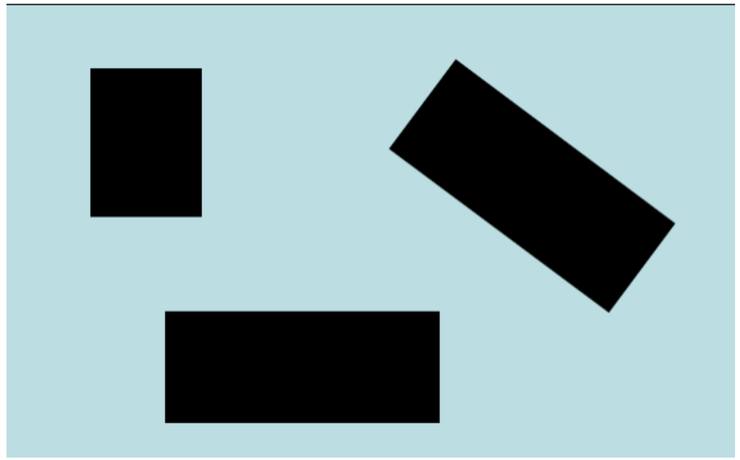
Fundamental Concept: C-space





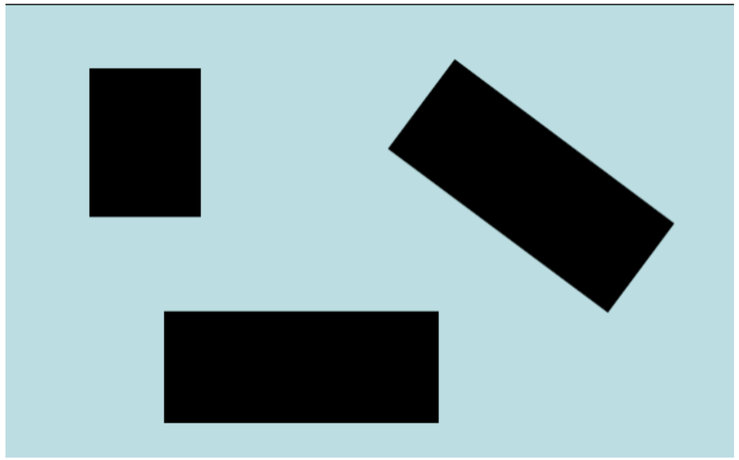
Howie Choset with slides from G.D. Hager, Z. Dodds, Dinesh Mocha, Ken Goldberg

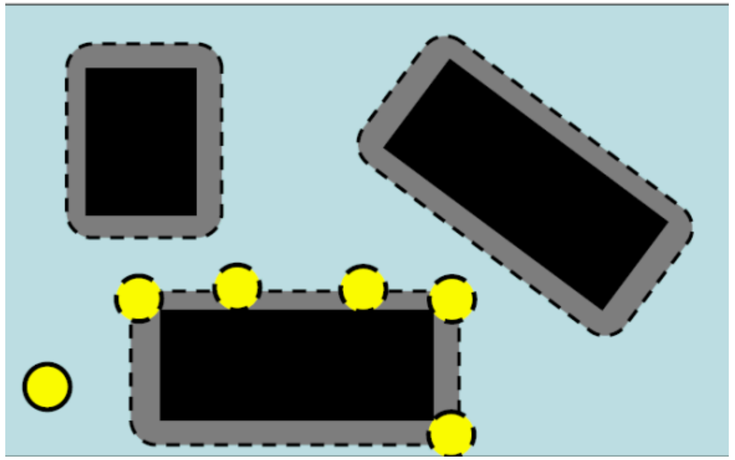
Fundamental Concept: C-space



Howie Choset with slides from G.D. Hager, Z. Dodds, Dinesh Mocha, Ken Goldberg

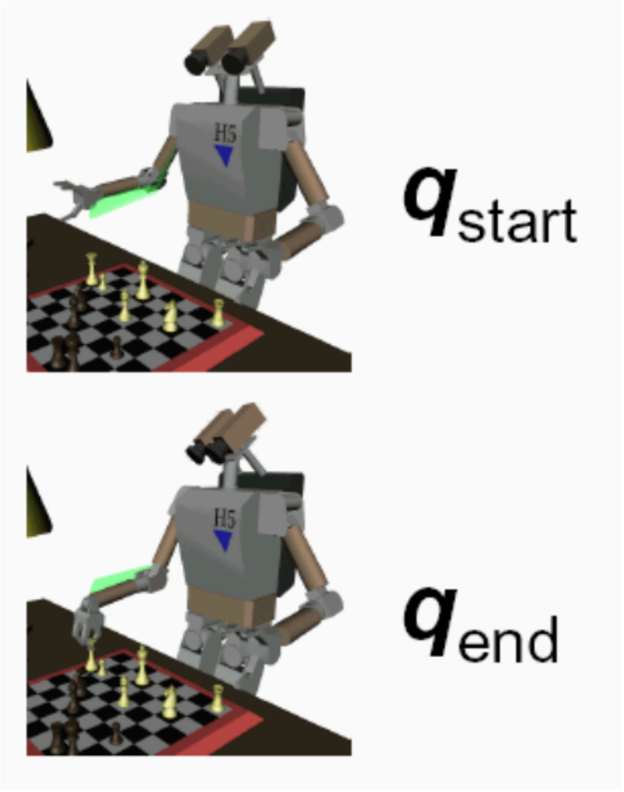
Fundamental Concept: C-space





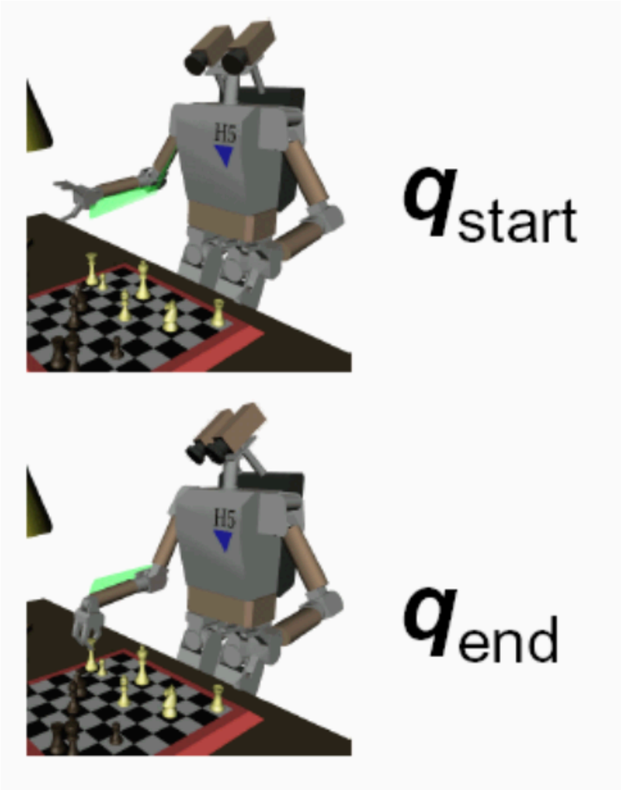
Howie Choset with slides from G.D. Hager, Z. Dodds, Dinesh Mocha, Ken Goldberg

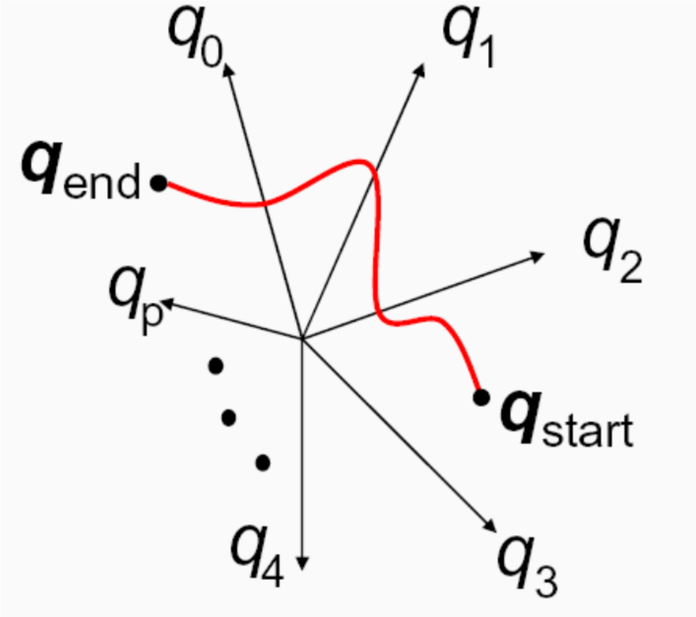
The motion planning problem



Slide credits: Howie Choset

The motion planning problem





Slide credits: Howie Choset

Formal guarantees of motion planning?

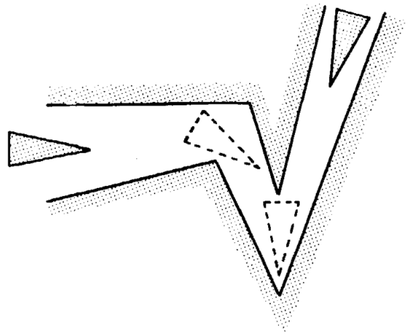
Communications on
PURE AND APPLIED MATHEMATICS

Article | Full Access

On the "piano movers'" problem I. The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers

Jacob T. Schwartz, Micha Sharir

First published: May 1983 | https://doi.org/10.1002/cpa.3160360305 | Citations: 299



Slide credits: Howie Choset

Formal guarantees of motion planning?

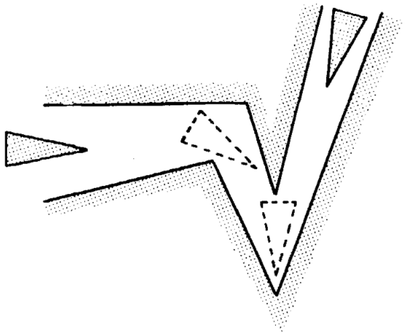
Communications on
PURE AND APPLIED MATHEMATICS

Article | Full Access

On the "piano movers'" problem I. The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers

Jacob T. Schwartz, Micha Sharir

First published: May 1983 | https://doi.org/10.1002/cpa.3160360305 | Citations: 299

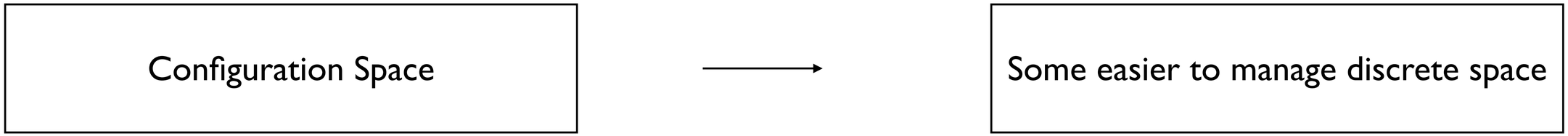


- Formal Result (but not terribly useful for practical algorithms):
  - *p*: Dimension of $\mathcal{C}$
  - *m*: Number of polynomials describing $\mathcal{C}_{free}$
  - *d*: Max degree of the polynomials
- A path (if it exists) can be found in time *exponential* in *p* and *polynomial* in *m* and *d*

[From J. Canny. "The Complexity of Robot Motion Planning Plans". MIT Ph.D. Dissertation. 1987]

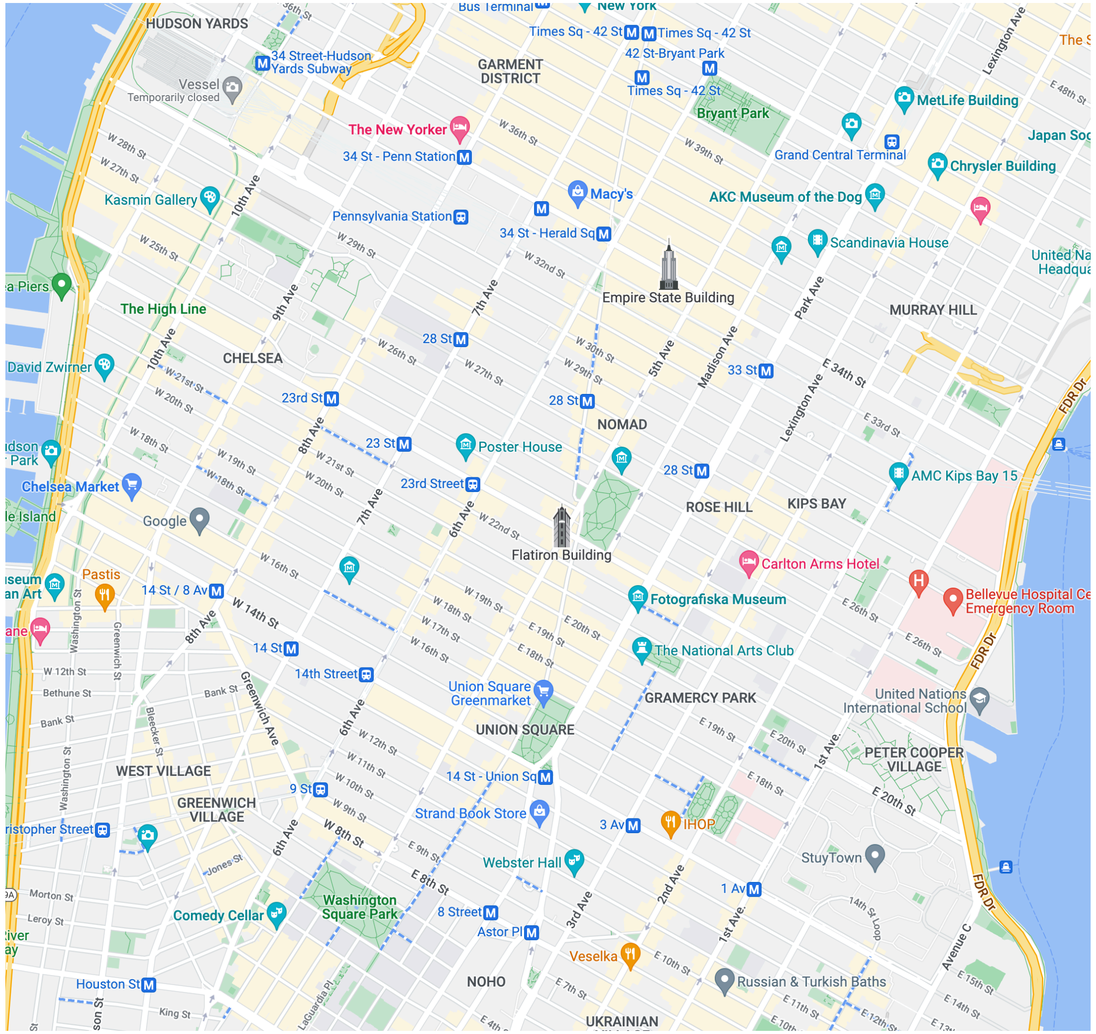
Slide credits: Howie Choset

What is done in practice?



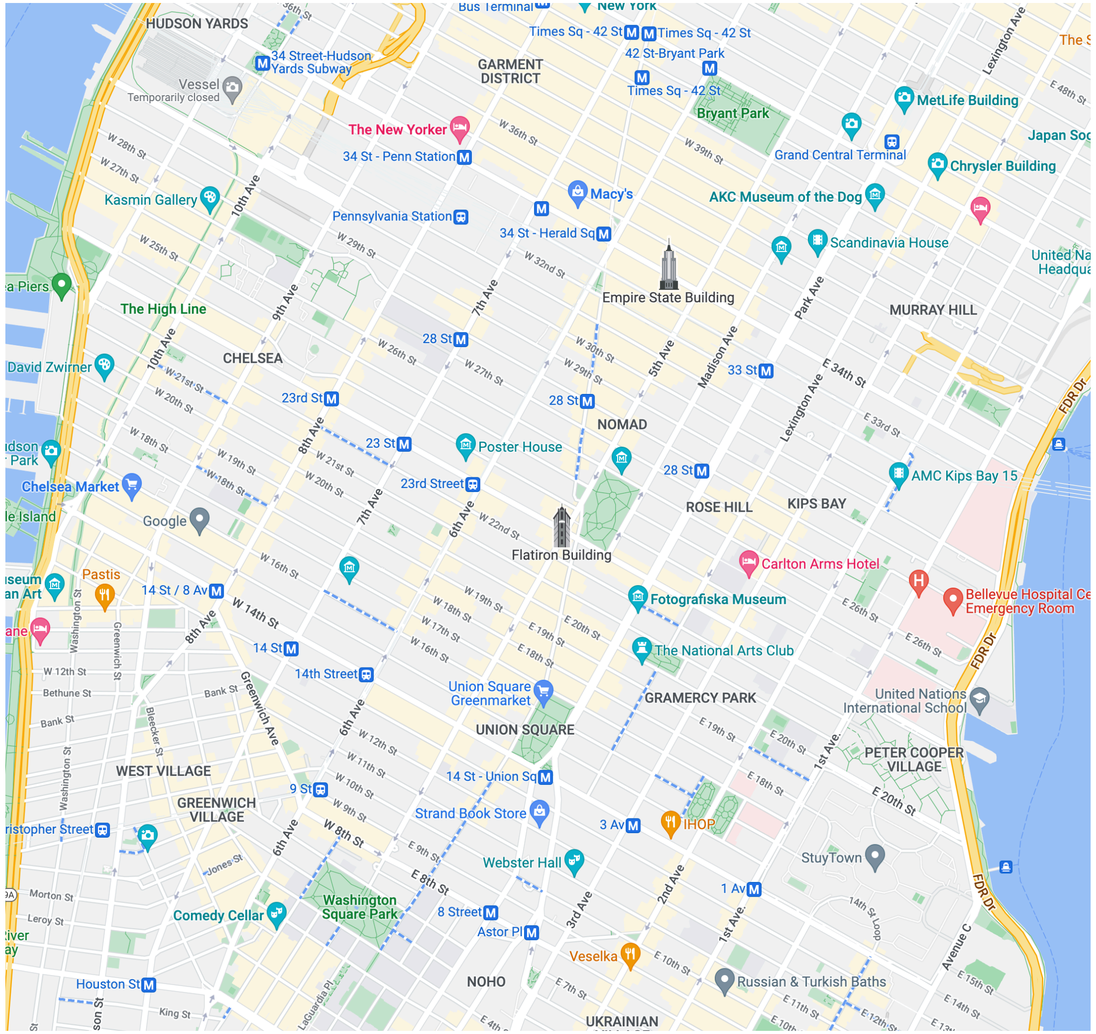
Slide credits: Howie Choset

Example



Slide credits: Howie Choset

Roadmaps

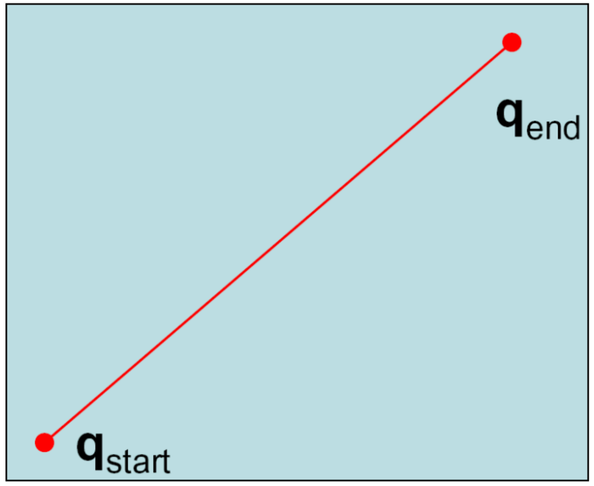


General Strategy:

1. Avoid searching the whole space.
2. Pre-compute a small graph that is guaranteed to be obstacle free (aka Roads).
3. Find a path from A to B using the road map.

Slide credits: Howie Choset

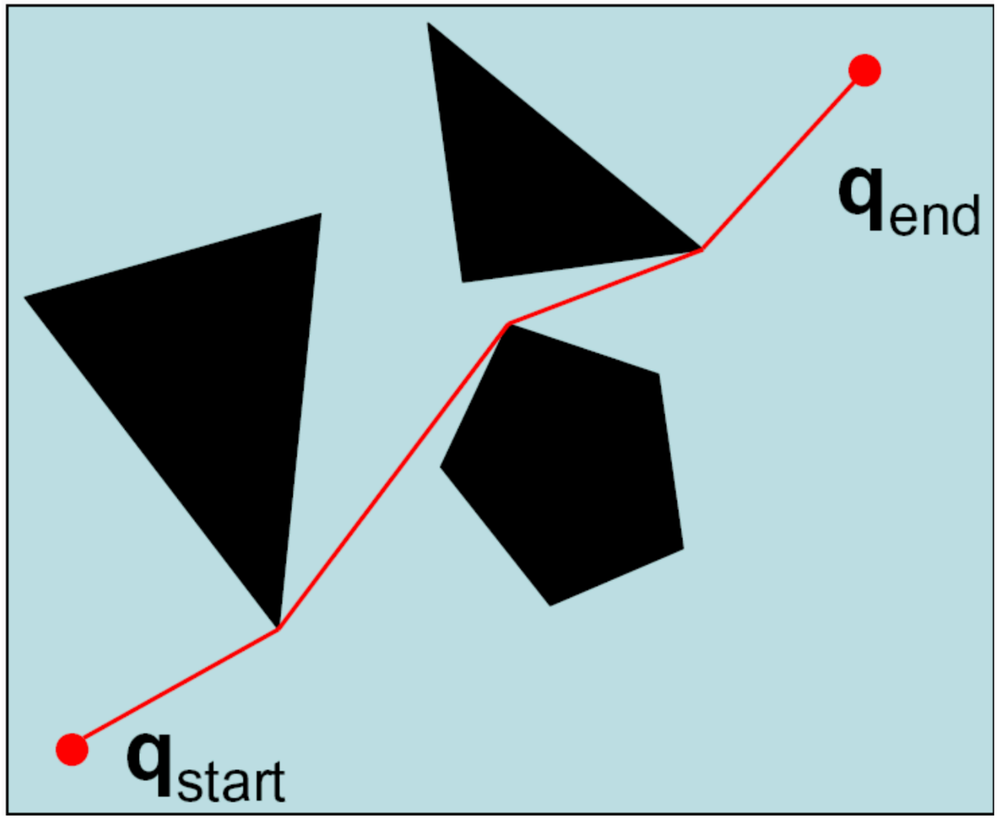
Visibility Maps



In the absence of obstacles, the best path is the straight line between $\mathbf{q}_{start}$ and $\mathbf{q}_{goal}$

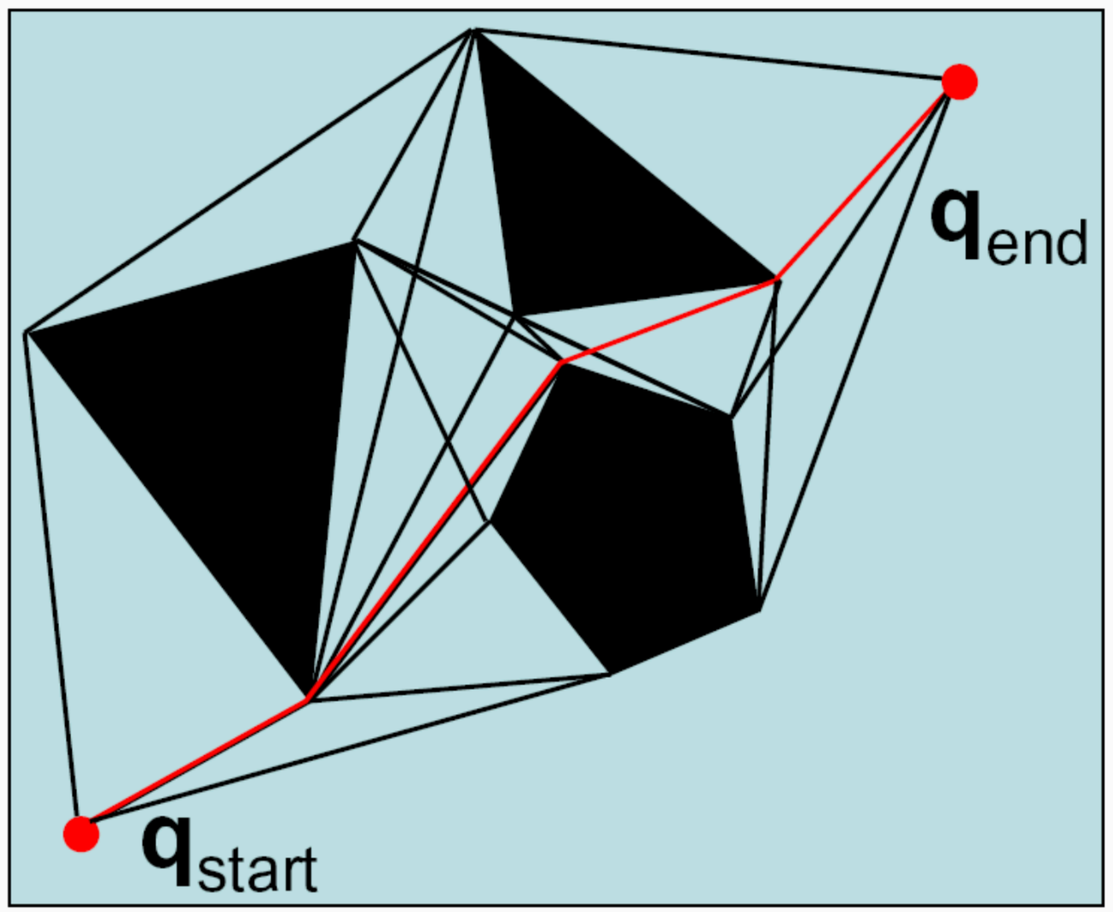
Slide credits: Howie Choset

Visibility Maps



Slide credits: Howie Choset

Visibility Maps

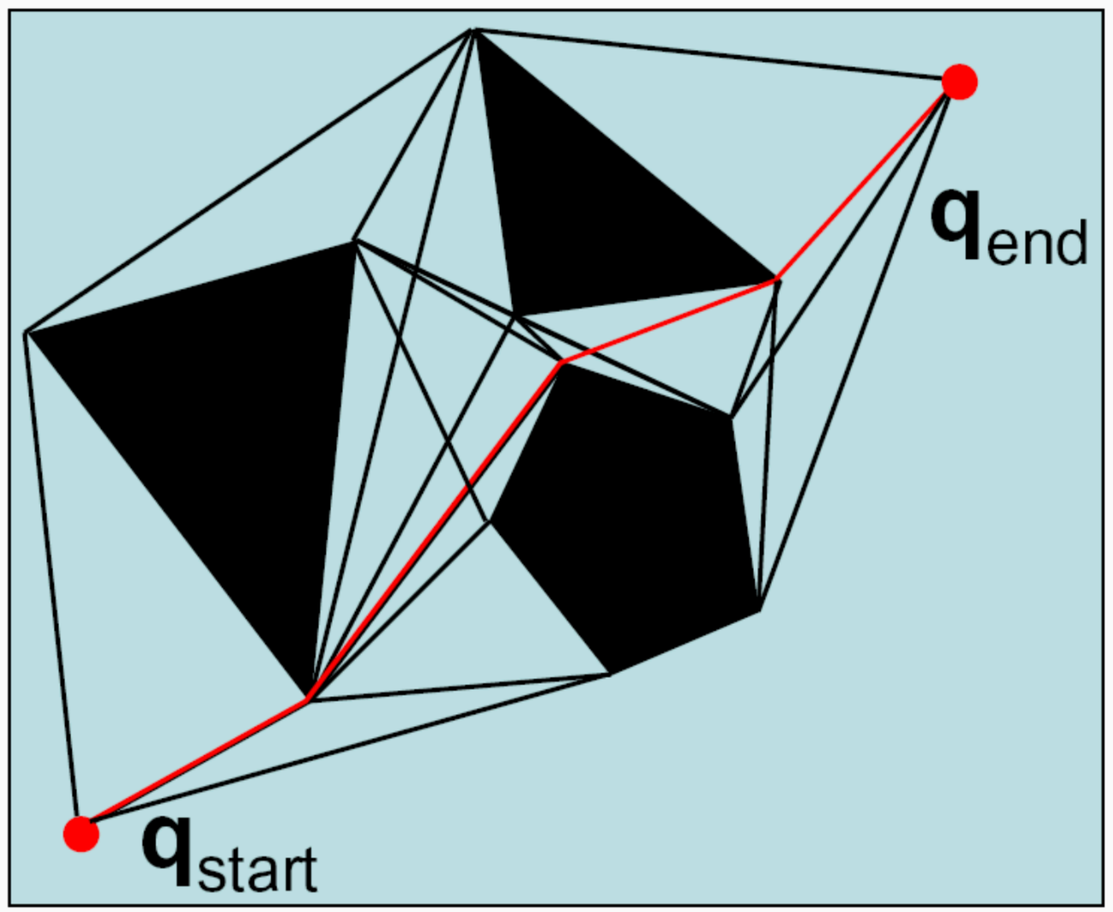


General Strategy:

1. Convert C-space to a graph.
2. Each polygonal vertex is a node.
3. Edges can be determined by ‘visibility’.
4. Once converted to graph… this is a graph search problem.

Slide credits: Howie Choset

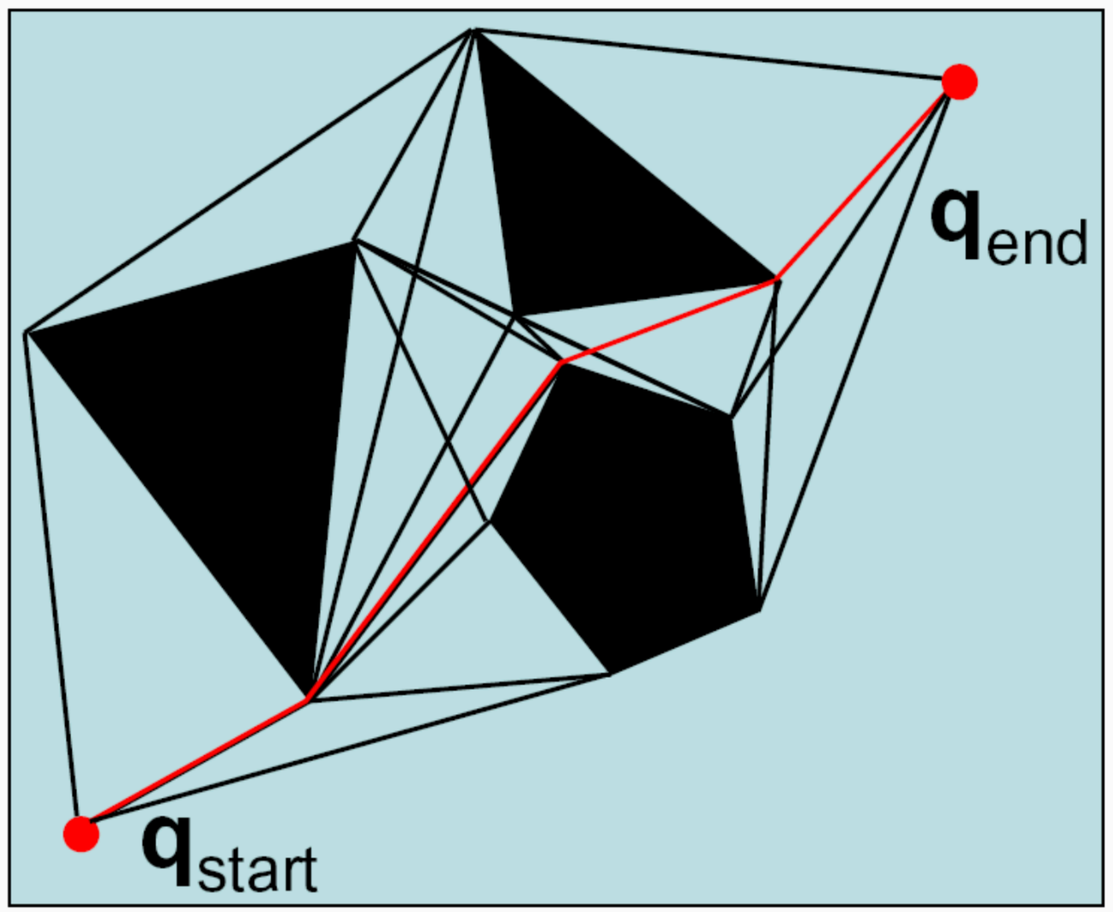
Visibility Maps



- N = total number of vertices of the obstacle polygons
- Naïve: $O(N^3)$
- Sweep: $O(N^2 \log N)$
- Optimal: $O(N^2)$

Slide credits: Howie Choset

Visibility Maps

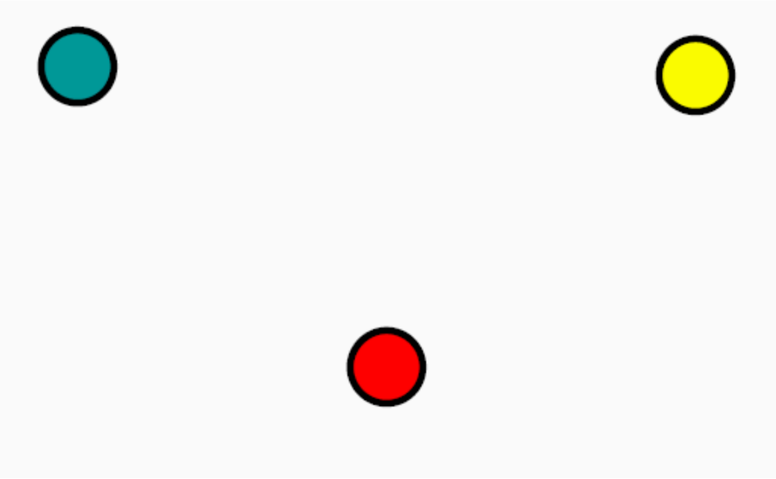


- N = total number of vertices of the obstacle polygons
- Naïve: $O(N^3)$
- Sweep: $O(N^2 \log N)$
- Optimal: $O(N^2)$

Apart from computational complexity, what is the issue here?

Slide credits: Howie Choset

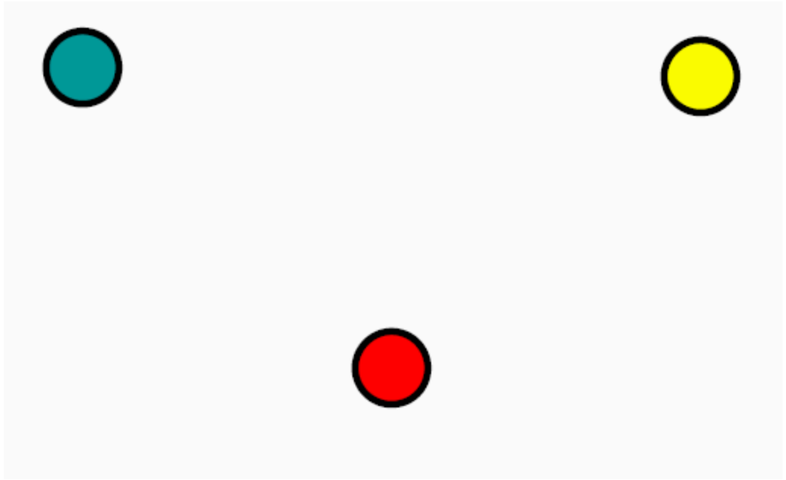
Voronoi Diagram



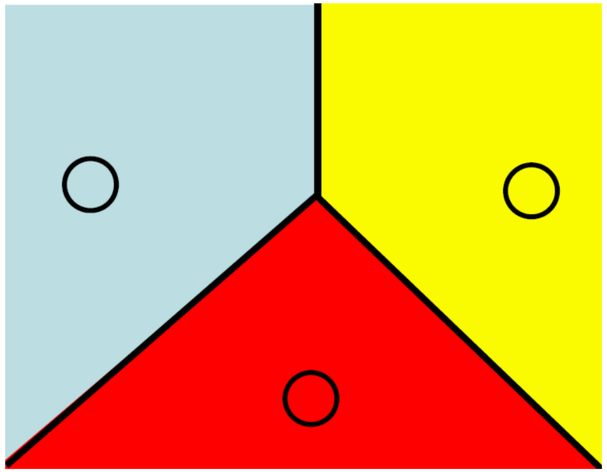
- Given a set of data points in the plane:
  - Color the entire plane such that the color of any point in the plane is the same as the color of its nearest neighbor

Slide credits: Howie Choset

Voronoi Diagram



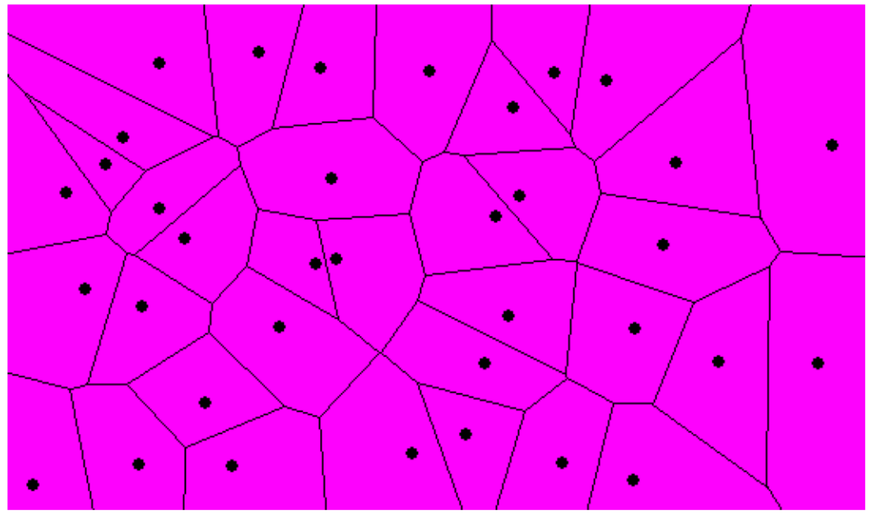
- Given a set of data points in the plane:
  - Color the entire plane such that the color of any point in the plane is the same as the color of its nearest neighbor



• Voronoi diagram = The set of line segments separating the regions corresponding to different colors

Slide credits: Howie Choset

Voronoi Diagram

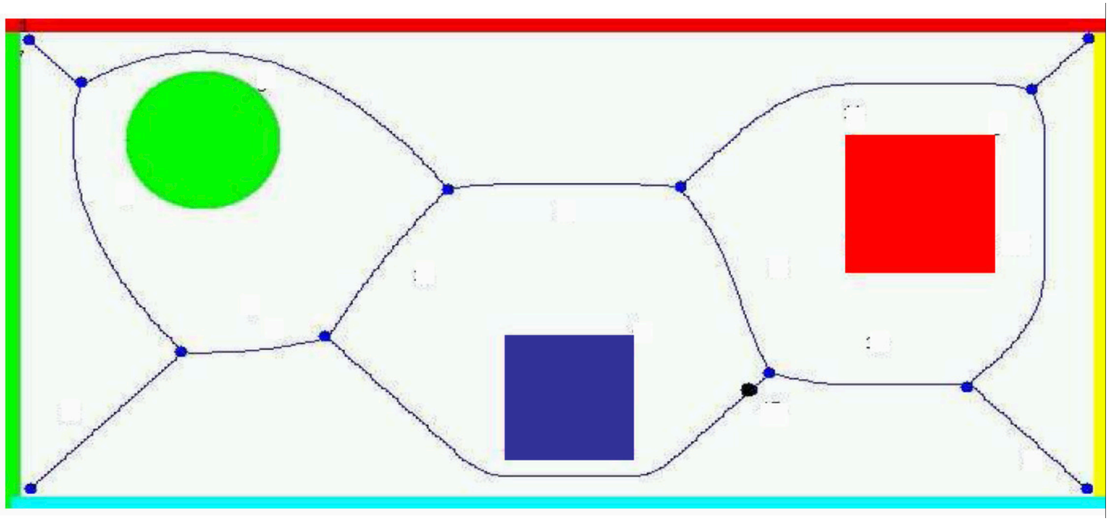


- Complexity (in the plane):
- O(*N* log *N*) time
- O(*N*) space

See http://alexbeutel.com/webgl/voronoi.html for an interactive demo

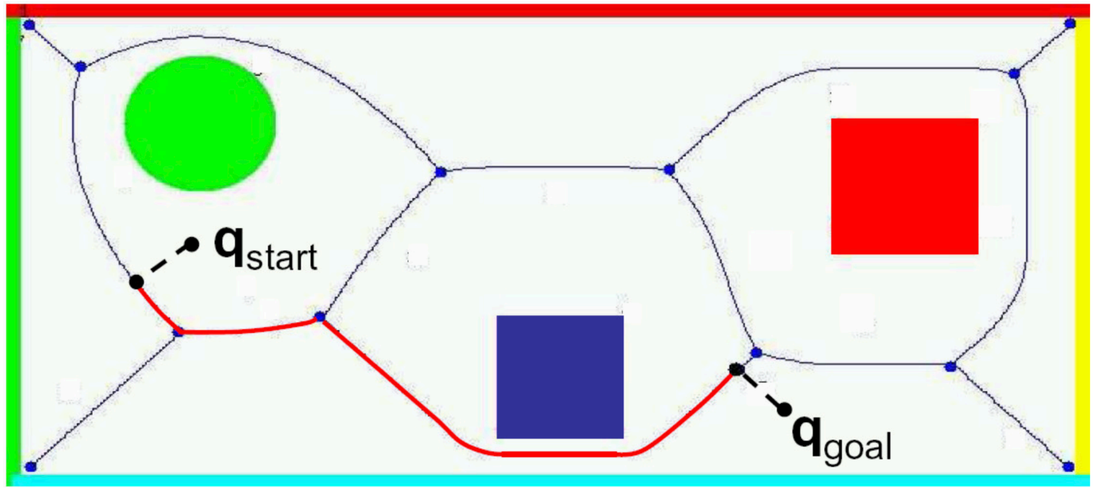
Slide credits: Howie Choset

Voronoi Diagram



Slide credits: Howie Choset

Voronoi Diagram

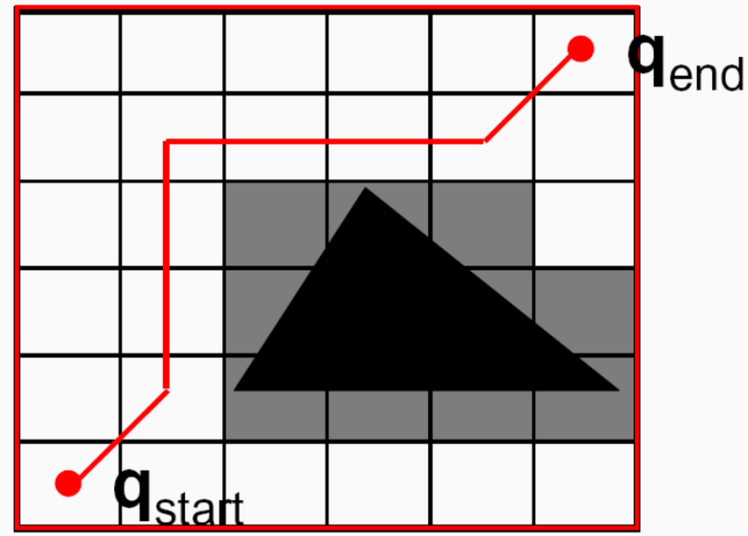


General Strategy:

1. Convert C-space to a Voronoi Diagram.
2. Given a start and end position, find closest points on the roadmap.
3. Solve motion planning as Graph search.

Slide credits: Howie Choset

Cell Decomposition

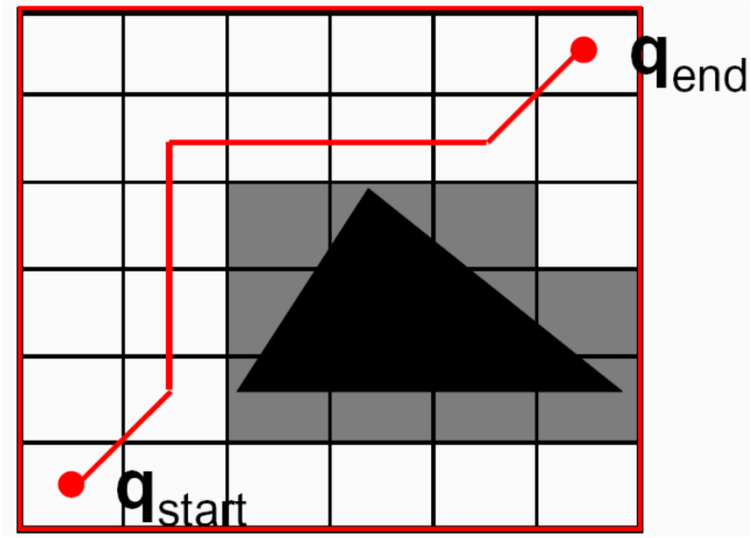


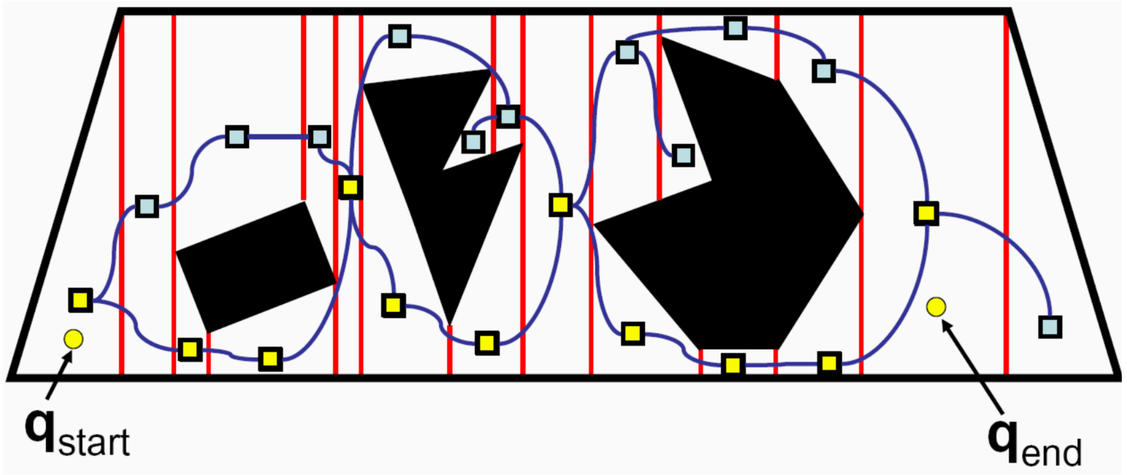
General Strategy:

1. Convert C-space to a Cells.
2. View each cell as a graph node.
3. Solve motion planning as Graph search.

Slide credits: Howie Choset

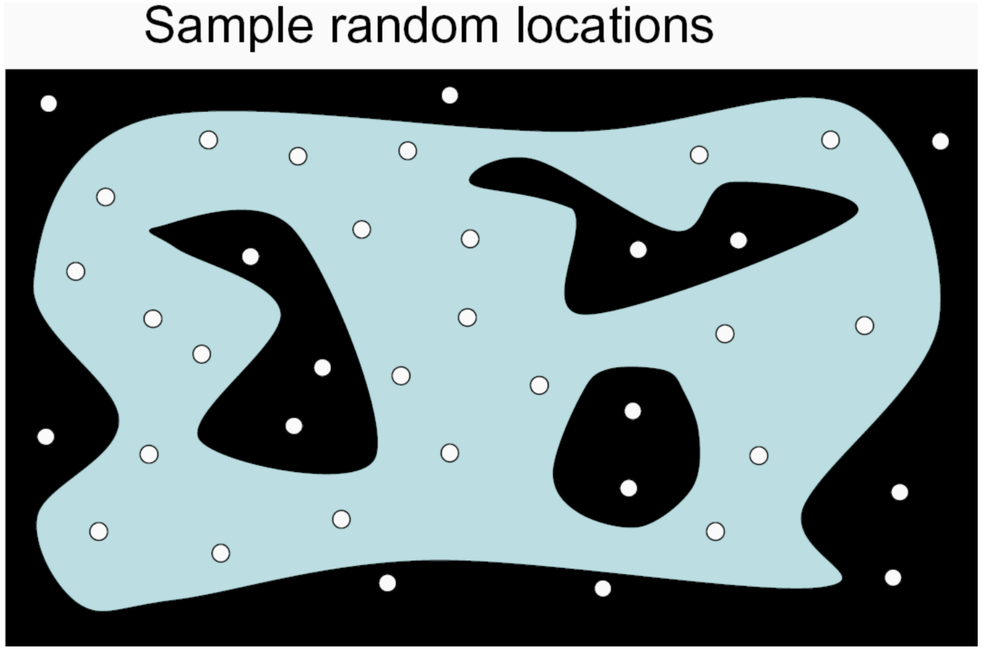
Cell Decomposition





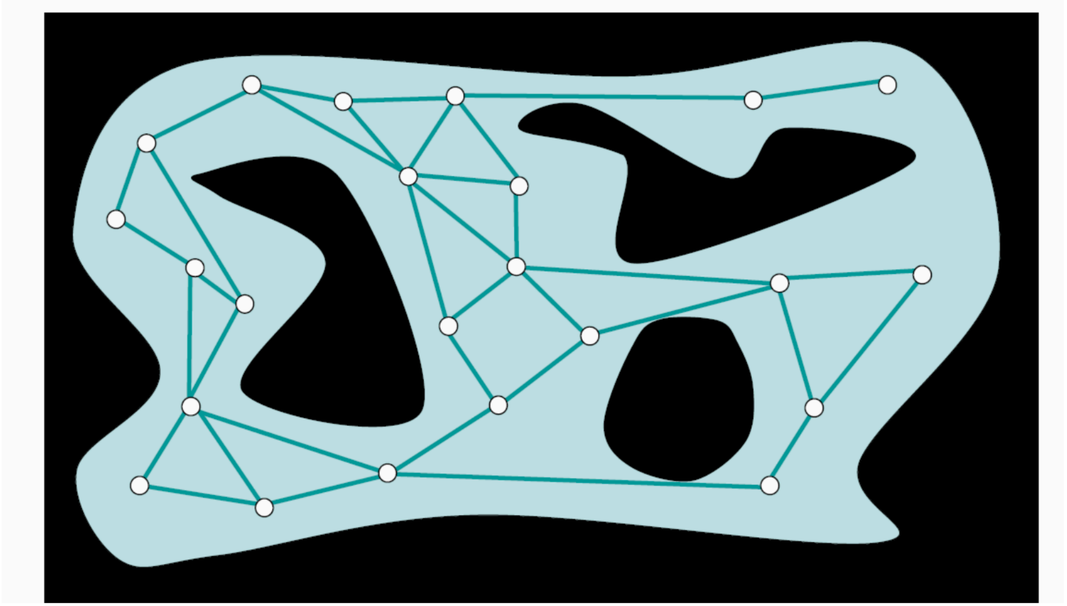
Slide credits: Howie Choset

Sampling based



Slide credits: Howie Choset

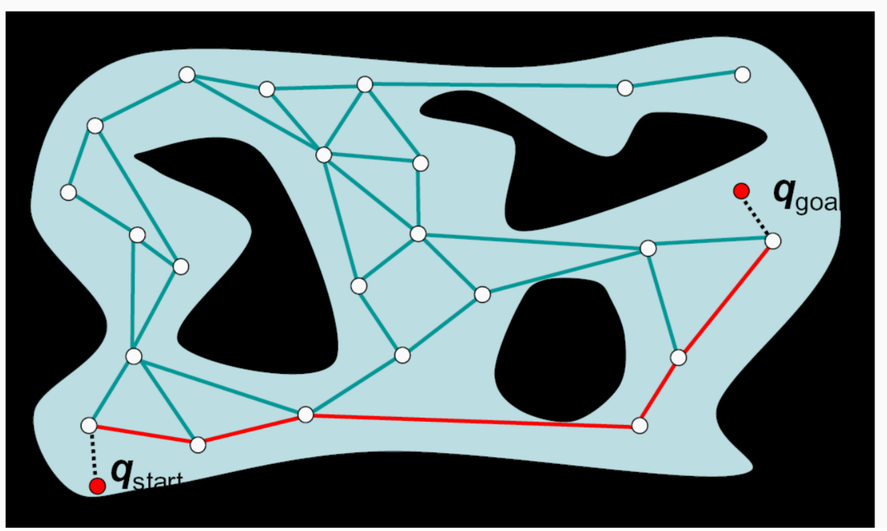
Sampling based



The resulting graph is a *probabilistic roadmap (PRM)*

Slide credits: Howie Choset

Sampling based



General Strategy:

1. Use random sampling to find points.
2. Create a graph from the sampled points.
3. Solve motion planning as Graph search.

Slide credits: Howie Choset